Machine Learning @ BECONA

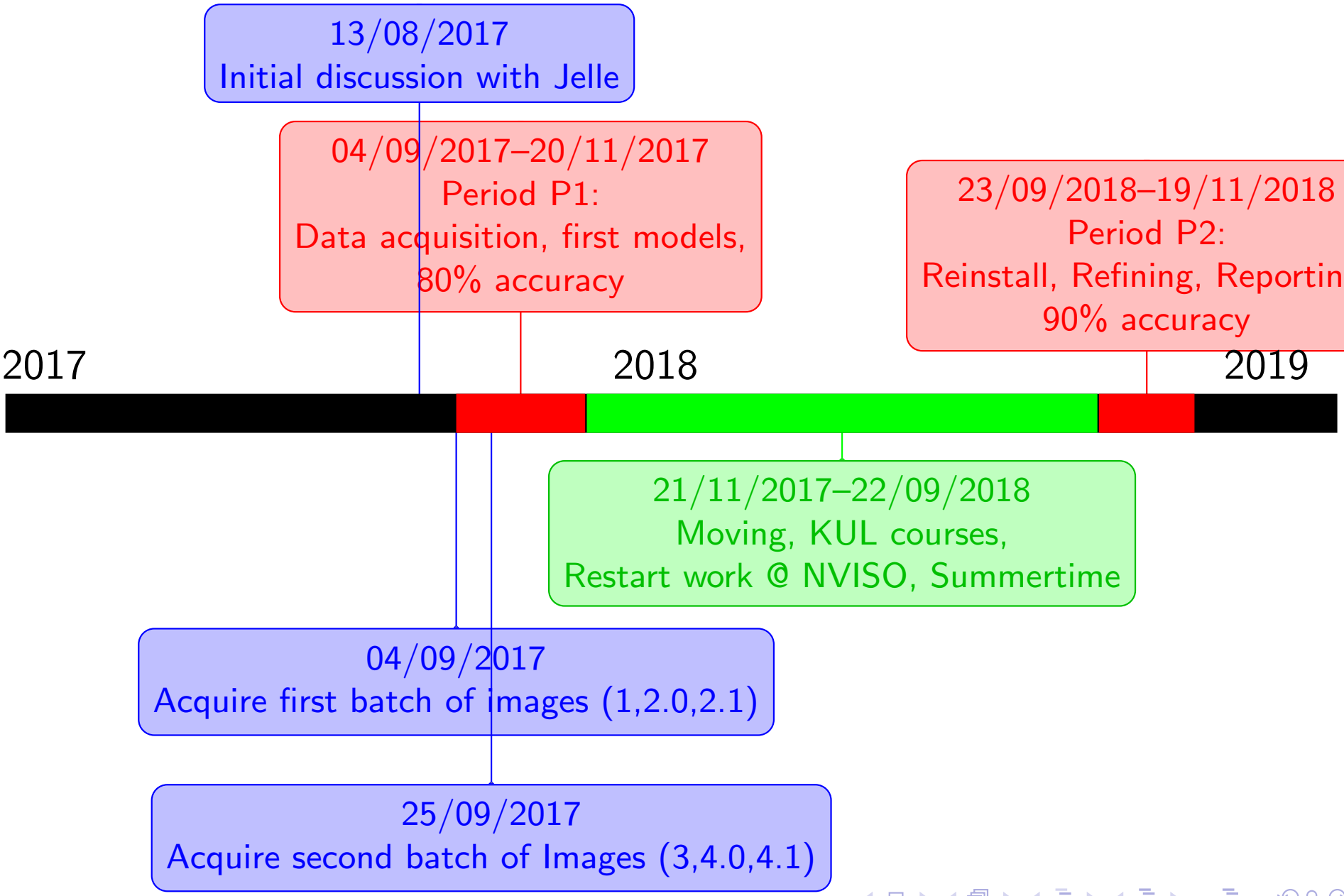
Proof of Concept Image Classification for Concrete Casting Equipment

Dieter Castel

September 20, 2019

Overview

Timeline


13/08/2017
Initial discussion with Jelle
04/09/2017–20/11/2017
Period P1:
Data acquisition, first models,
80% accuracy
23/09/2018–19/11/2018
Period P2:
Reinstall, Refining, Reporting
90% accuracy
2017
2018
2019
21/11/2017–22/09/2018
Moving, KUL courses,
Restart work @ NVISO, Summertime
04/09/2017
Acquire first batch of images (1,2.0,2.1)
25/09/2017
Acquire second batch of Images (3,4.0,4.1)

Problem Definition

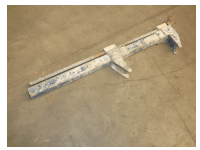
Problem Definition

- Automated Sorting of Becona Rental Items Equipment
  - Exploratory Proof of Concept
  - Focused on Object Recognition Software System
  - Educational Machine Learning project for myself

Q:Is machine learning a viable approach to be used as classification component in the automated sorting machine?

Items

About 30 items exist, for the proof-of-concept I focused on a subset of 6:

Id	Name	# images	example
1	Spanklem	223	
2.0	Vleugelmoer Opleg Recht - Oud	208	
2.1	Vleugelmoer Opleg Recht - Nieuw	244	
3	Vleugelmoer Opleg Rond	238	
4.0	Variable Spanklem - Kort	270	
4.1	Variable Spanklem - Lang	251	

Machine learning

- Can solve problems programmers can't program (too complex, too much data, too little time).
- Combine data and algorithms to teach the machine to solve the problem.
- Goal is predictive power = Generalisation = Learning

Machine learning solution consists of three major components

Representation How to represent the model: what input, what output, model structure, what modelling language,...

Evaluation A function is needed to evaluate and compare models.

Optimization How to search for the best scoring model?

Representation

First considered classification of objects by weight:

- Upside: Much Simpler model: Linear or SVM classifier
- Downside: **Overlapping weights**: 0,6kg, 5,3kg, 0,16kg, 6,2kg for several objects
  AND likely influence on weight of:
  - Concrete Remains
  - Wear
  - make variability
  - ...

Representation

Not all items can be classified by weight...

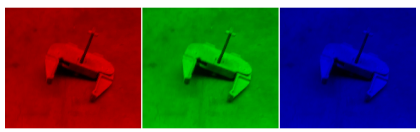
Representation

Image classification seemed good (and interesting) candidate:
Best image classification systems currently use machine learning, more specifically:

- Supervised machine learning (labeled data).
- Convolutional Neural Networks (CNNs).
- Color(RGB) Image as input.

Q: Are CNNs a viable solution for Becona rental items classification?

Convolutional Neural Network structure (simplified)



Input layer

Hidden layers

Output layer

R-Pixel (0,0)

G-Pixel (0,0)

B-Pixel (0,0)

R-Pixel (1,0)

$P(1)$

$P(2.0)$

$P(2.1)$

$P(3)$

$P(4.0)$

$P(4.1)$

Approach & Lessons Learned

- Training an image classification Neural Network from scratch needs **millions** of images and days to weeks of computation time
- In two afternoons I took merely 1434 pictures and don't have a datacenter.
- **solution**: stand on the shoulders of giants → **transfer learning**
  - Start from Google-trained InceptionV3 & Xception Neural Networks
  - NN weights trained for IMAGENET-1000 project: millions of tagged images.
  - These networks can categorize 1000 common items very well (80% accurate)
  - Fine-tune Neural Network on this specific data set starting with the basemodel weights.

Image Augmentation

- Still 1434 pictures is a small dataset for transfer learning with CNNs.
- **solution:** Image Augmentation - Random modifications of the dataset
  - Zoom in
  - Flip image horizontally
  - Flip image vertically
  - shift image along x axis
  - shift image along y axis
  - rotate image
  - Make red, green or blue channel shifts

Image augmentation 1, 2.0, 2.1

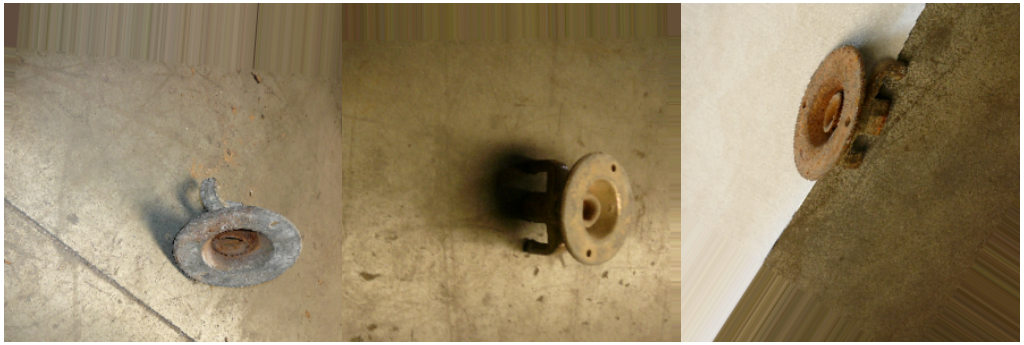
Item 1.0

Item 2.0

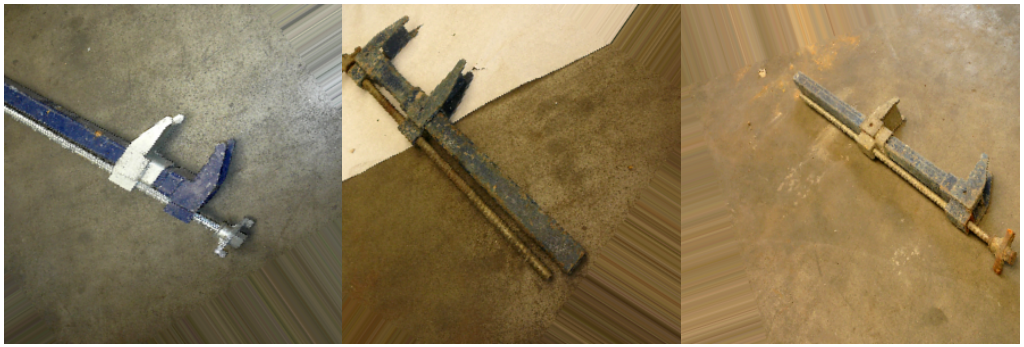
Item 2.1

Image augmentation 3, 4.0, 4.1

Item 1.0



Item 2.0



Item 2.1

Training CNNs is a probabilistic process

- Training Neural Networks is probilistic (stochastic) process, it contains random variables.
- Train the same model twice, it will have different results
- **How to compare different models?**

3 Machine Learning Components

Representation Supervised machine learning, Convolutional Neural Networks, RGB image as input.

Evaluation Categorical Crossentropy, **5-fold Cross Validation**

Optimization Stochastic Gradient Descent, **Trial & Error**/Intuition (aka Student Gradient Descent), RMSprop, Adam,...

LL:5-fold cross-validation

For each model run 5 experiments

And average over the 5 different experiments.

Hardware & Software Stack

	Period 1	Period 2
Keras	2.0.6	**2.2.2**
Tensorflow	1.3.0	**1.13.0**
Python	3.6	3.6
cuDNN	5.1	**7.0**
CUDA	8.0	**9.0**
Ubuntu	16.04.3 LTS	**fresh 16.04.3 LTS**
Storage	Seagate 3TB-7600RPM	**Samsung 265GB SSD**
NVIDIA GPU	GTX 1060 6GB	
Intel CPU	Core i5-3570K CPU 3.40GHz	
RAM	16 GB	

Table: In P1 trained with testfiles on HDD, later in P2 reinstalled full software stack, with updated libraries, on new SSD

LL: Disk speed/Library version matters

Table: Rough estimate: Factor 10 training speed improvement in Period 2

Period	Experiment ID	Approximate Training Hours	Best Validation Loss (lower = better)
1	mXc_v5_split0	7	0.17
1	mXc_v6_split0	7	0.2
1	mXc_v5_split2	7	0.16
1	mXc_v6_split2	7	0.18
2	mXc_v5_split0	0.6666	0.25
2	mXc_v6_split0	0.6500	0.22
2	mXc_v5_split2	0.5833	0.2
2	mXc_v6_split2	0.6666	0.22

LL: Signal/Noise matters

- If $\frac{signal}{noise}$ of image is too small, CNNs have hard time picking up on the signal.
- I learned it the hard way:
  - First trained model with 675 uncropped rectangular images.
    $\Rightarrow$ Poor training results.
  - After 1,5h of cropping 675 images and retraining the network using Cropall tool
    $\Rightarrow$ Better results 80% accuracy?

LL: Signal/Noise matters

uncropped

cropped

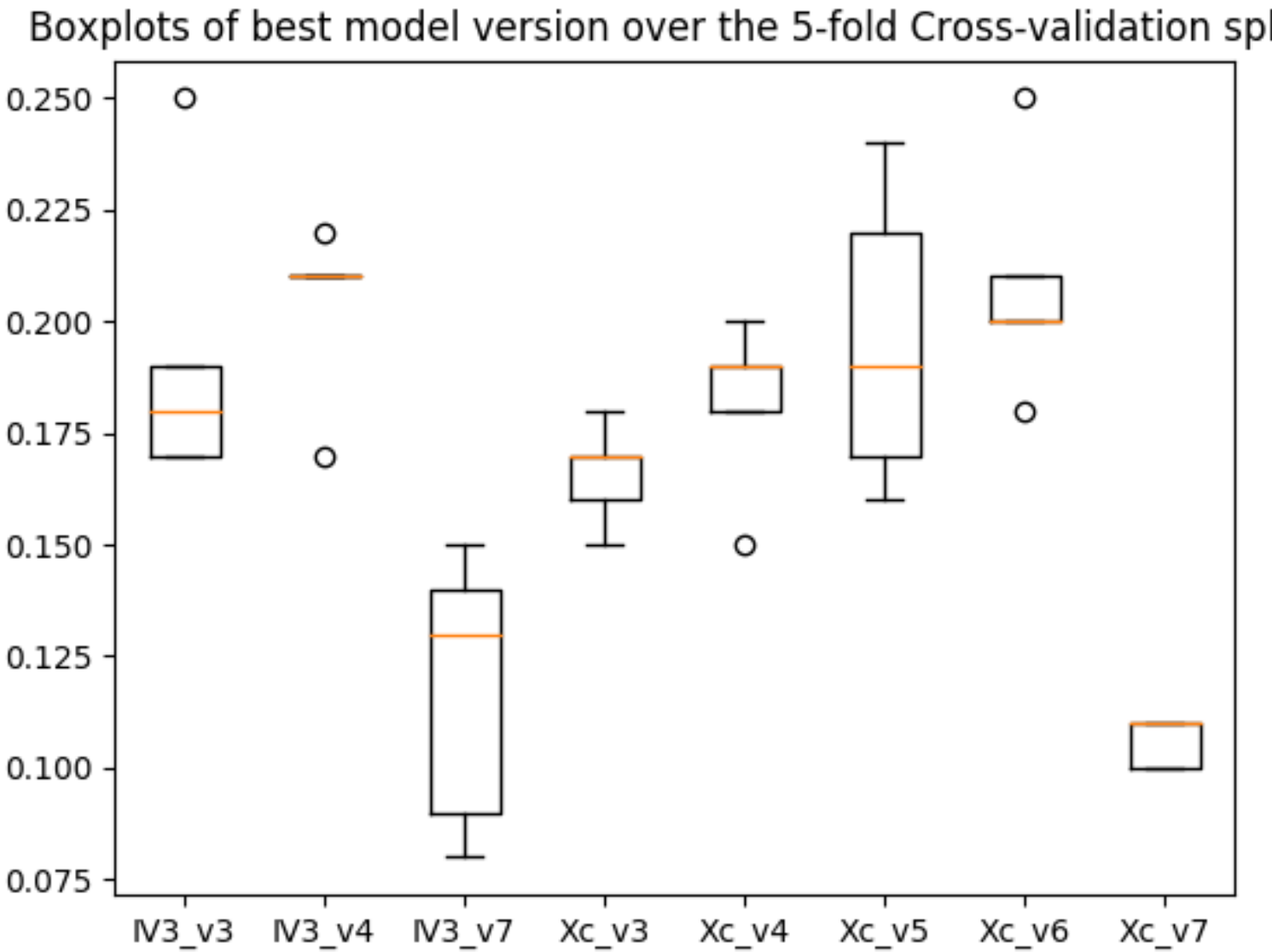
LL: I learned a lot!

- Setting up a Graphics Processing Unit CNN training machine (twice).
- Using state of the art CNN libraries
- Improved python scripting
- Learned a lot about Neural Networks and Machine Learning
- Read many academic papers, books, blogs, ...

For me the project was a huge success!

Results

Validation Loss



basemodel: IV3_ = InceptionV3, XC_ = Xception

Best Model

- Best model is Xc_v7
  - Has Xception with ImageNet weights as basemodel.
  - Fully-connected penultimate layer of 256 neurons.
  - Was optimized with Stochastic Gradient Descent first and the Adam optimizer finally.
- Has average validation loss of 0.1011802
  $\Rightarrow$ validation accuracy of $1 - 0.1011802 = 0.8988198 \approx 90\%$

Q: Are CNNs a viable solution for Becona rental items classification? YES... but

Caveat of this Proof of Concept

Note that while 90% accuracy seems impressive this is still a very limited dataset. There is no guarantee that the model will perform as well with images taken today. Models CAN degrade over time and are biased by the dataset they are trained on.

Looking Forward

Advice

- Create Data (**Image**,**weight**,...) Capture Pipeline
  - Combining sensors will often help (= Ensemble Model).
- For CNNs: more data, better results
- Multiple-camera's $\rightarrow$ vote per camera
  - Upside: Will likely improve accuracy, more camera's == more data
  - Downside: Processing multiple images fast enough might be a challenge
- Deploy current best model(s) and monitor real world test performance on a larger dataset.

Costs and tradeoffs to consider vs human classification

- Financial cost
- Sorting Speed (vs 24/7)
- IT setup cost
- Long-term cost hard to estimate
  - Software and model maintenance
  - Model degradation?
- Possible Brain drain cost
- Physical effort
- Classification accuracy

CNN Intresting topics for future research

- Adding different sensors (Depth sensor, Non-visible spectra) will likely improve accuracy
  - Upside: Will likely improve accuracy
  - Downside: Transfer learning not yet possible, long training times, $\frac{signal}{noise}$?
- Exploring 3D renderings of the CAD files could help recognition.
  - Even low fidelity renderings have shown to help train Neural Networks
  - Upside: CAD files of all items exist already.
  - Downside: Rendering also costs computation time.
- Generate more data synthetically using Generative Adverserial Neural Networks
  - Advanced form of image augmentation
  - Upside: Could significantly improve robustness and accuracy
  - Downside: Technically challenging, high computation time cost

Possible collaborations

Consider collaborations.

- Internship students could help create the machine learning pipeline.
- University partnerships with industry is highly encouraged (master thesis?)
- Me (I'm looking for a new job): as doctoral student, as consultant (Kunlabora, Flanders make, ...), directly?

Q&A

Questions for Becona

- How accurate do the humans sorters sort?
- How much items are there in total? (How much % did I photograph now?)
- How much items are there per item class? (Knowing class inbalance can help model)
- Can I use this as portfolio for my job search?

Bibliography

- Blogs, The Internet, Keras, Tensorflow docs, Open-source libraries,...
- Rules of Machine Learning: Best Practices for ML Engineering
- Paper: A Few Useful Things to Know about Machine Learning
- Book: Pattern Recognition and Machine Learning - Christopher Bishop
- Book: Deep learning - Ian Goodfellow and Yoshua Bengio and Aaron Courville
- Image from researchgate.net